

[> home](#) [> about](#) [> feedback](#) [> login](#)

US Patent & Trademark Office

Citation

International Conference on Mobile Computing and Networking [>archive](#)
Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking [>toc](#)
1999 , Seattle, Washington, United States

Next century challenges: data-centric networking for invisible computing: the Portolano project at the University of Washington

Authors

Mike Esler
Jeffrey Hightower
Tom Anderson
Gaetano Borriello

Sponsors

SIGCOMM : ACM Special Interest Group on Data Communication
SIGMOBILE : ACM Special Interest Group on Mobility of Systems, Users, Data and Computing
SIGMETRICS : ACM Special Interest Group on Measurement and Evaluation
SIGOPS : ACM Special Interest Group on Operating Systems
SIGMOD : ACM Special Interest Group on Management of Data


Publisher

ACM Press New York, NY, USA

Pages: 256 - 262 Series-Proceeding-Article

Year of Publication: 1999

ISBN:1-58113-142-9


 <http://doi.acm.org/10.1145/313451.313553> (Use this link to Bookmark this page)

[> full text](#) [> references](#) [> citings](#) [> index terms](#) [> peer to peer](#)


[> Discuss](#)

[> Similar](#)

[> Review this Article](#)

 [Save to Binder](#)

[> BibTex Format](#)

[↑ FULL TEXT:](#)  [Access Rules](#)

 pdf 1.03 MB

[↑ REFERENCES](#)

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the

complete List rather than only correct and linked references.

- 1 G. Abowd and C. Atkeson. Future computing environments: Cyberdesk. Technical report, Georgia Institute of Technology, 1998. <http://~.~.gat~ch~du/f/~yb~rd~k/>.
- 2 Pai Chou , Ross Ortega , Ken Hines , Kurt Patridge , Gaetano Borriello, ipChinook: an integrated IP-based design framework for distributed embedded systems, Proceedings of the 36th ACM/IEEE conference on Design automation conference, p.44-49, June 21-25, 1999, New Orleans, Louisiana, United States
- 3 Douglas E. Comer, Internetworking with TCP/IP (2nd ed.), vol. I, Prentice-Hall, Inc., Upper Saddle River, NJ, 1991
- 4 World Wide Web Consortium. Extensible markup language (XML), 1998. <http://www.wc3.org/XML/>.
- 5 S. Czerwinski, T. Hodes, B. Zhao, and A. Joseph. The service discovery service: Searching for computation power, 1999. <http://www.cs.berkeley.edu/czerwin/sdsproject.html>.
- 6 Nigel Davies , Adrian Friday , Gordon S. Blair , Keith Cheverst, Distributed systems support for adaptive mobile applications, Mobile Networks and Applications, v.1 n.4, p.399-408, Dec. 1996
- 7 W. Keith Edwards , Elizabeth D. Mynatt , Karin Petersen , Mike J. Spreitzer , Douglas B. Terry , Marvin M. Theimer, Designing and implementing asynchronous collaborative applications with Bayou, Proceedings of the 10th annual ACM symposium on User interface software and technology, p.119-128, October 14-17, 1997, Banff, Alberta, Canada
- 8 O. Etzioni, \$. Hanks, and D. Weld et al. Internet softbot research, 1998. <http://www.cs.washington.edu/research/projects/softbots/www/softbots.html>.
- 9 Tim Finin. UMBC agent web, 1998.
- 10 K. P. Fishkin, T. P. Moran, and B. L. Harrison. Embodied user interfaces: Towards invisible user interfaces. Technical report, Xerox Palo Alto Research Center, Palo Alto, CA, 1998.
- 11 HomeRF Working Group. HomeRF, 1999. <http://www.homerf.org/>.
- 12 Object Management Group. Corba news, 1998. <http://www.omg.org/corba/>.
- 13 J. Hartman, U. Manber, L. Peterson, and T. Proebsting. Liquid software: A new paradigm for networked systems. Technical report, University of Arizona, Tuscon, AZ, 1996.
- 14 The HAVi specification. Technical report, Home Audio/Visual Interoperability Consortium, 1998. <http://www.havi.org/>.
- 15 T. D. Hodes and R. H. Katz. Enabling "smart spaces:" entity description and user interface generation for a heterogeneous component-based distributed system. In DARPA/NIST Smart Spaces Workshop, Gaithersburg, Maryland, 1998. DARPA/NIST.
- 16 Todd D. Hodes , Randy H. Katz , Edouard Servan-Schreiber , Lawrence Rowe, Composable ad-hoc mobile services for universal interaction, Proceedings of the third annual ACM/IEEE international conference on Mobile computing and networking, p.1-12, September 26-30, 1997, Budapest, Hungary
- 17 S. Kent and R. Atkinson. Security architecture for the internet protocol, 1998. RFC 2401.
- 18 A. J. Kunzman and A. T. Wetzel. 1394 high performance serial bus: The digital interface for ATV. IEEE Transactions on Consumer Electronics, 14(13):893- 900, 1995.
- 19 Microsoft. Microsoft COM technologies, 1998. <http://www.microsoft.com/com/>.
- 20 Sun Microsystems. The NFS distributed file system. Technical report, Palo Alto, CA, 1995.
- 21 Motorola. Motorola VoxML, 1998. <http://voxml.motorola.com/>.
- 22 Donald A. Norman, The invisible computer, MIT Press, Cambridge, MA, 1998

- 23 H. S. Nwana. Software agents: an overview. *Knowledge Engineering Review*, 1(3):205-244, 1996.
- 24 File systems in a distributed computing environment. Technical report, Open Software Foundation, 1991.
- 25 C. Perkins. SLP white paper. Technical report, Sun Microsystems, 1998. <http://playground.sun.com/srvloc/>.
- 26 C. E. Perkins. IP mobility support, 1996. RFC 2002.
- 27 Charles E. Perkins , Harry Harjono, Resource discovery protocol for mobile computing, *Mobile Networks and Applications*, v.1 n.4, p.447-455, Dec. 1996
- 28 Universal Plug and Play Forum. Universal plug and play. <http://www.upnp.org/>.
- 29 IBM Research. IBM T-Spaces project. <http://www,almaden.ibm.com/cs/TSpaces/>.
- 30 Frequently asked questions about today's cryptography. Technical report, RSA Data Security, Inc., San Mateo, CA, 1998.
- 31 Carnegie Mellon University School of Computer Science. Coda file system, 1999. <http://www.coda.cs.cmu.edu/>.
- 32 Bluetooth SIG. Bluetooth technology, 1999. <http://www.bluetooth.com/technology/>.
- 33 Universal serial bus specification, revision 1.1. Technical report, Compaq, Intel, Microsoft, and NEC, 1998. <http://www.usb.org/developers/data/usbll.pdf>.
- 34 J. Waldo. Jini architecture overview. Technical report, Sun Microsystems, Inc., Palo Alto, CA, 1998.
- 35 Roy Want , Kenneth P. Fishkin , Anuj Gujar , Beverly L. Harrison, Bridging physical and virtual worlds with electronic tags, *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, p.370-377, May 15-20, 1999, Pittsburgh, Pennsylvania, United States
- 36 R. Want, B. Schilit, N. Adams, R. Gold, D. Goldberg, K. Petersen, J. Ellis, and M. Weiser. The PARCTAB ubiquitous computing experiment. In T. Imielinski, editor, *Mobile Computing*, chapter 2, pages 45-101. Kluwer Publishing, February 1997.
- 37 A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. *IEEE Personal Communications*, 4(5):42-47, 1997.
- 38 Wireless data primer. Technical report, Wireless Data Forum, Washington, D.C., 1998.
- 39 M. Weiser. Open house. *Review*, 2, 1996.
- 40 D. J. Wetherall, J. V. Guttag, and D. L. Tennenhouse. ANTS: A toolkit for building and dynamically deploying network protocols. In *OPENARCH '98*, San Francisco, CA, 1998. IEEE.
- 41 L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE-Network*, 7(5):8-18, 1993.

↑ CITINGS 6

Andrew C. Huang , Benjamin C. Ling , John Barton , Armando Fox, Making computers disappear: appliance data services, *Proceedings of the seventh annual international conference on Mobile computing and networking*, p.108-121, July 2001, Rome, Italy

Yolanda Villate , Arantza Illarramendi , Evaggelia Pitoura, Keep your data safe and available while roaming, *Mobile Networks and Applications*, v.7 n.4, p.315-328, August 2002

Deepak Ganesan , Ramesh Govindan , Scott Shenker , Deborah Estrin, Highly-resilient, energy-efficient multipath routing in wireless sensor networks, *ACM SIGMOBILE Mobile Computing and Communications Review*, v.5 n.4, October 2001

Guruduth Banavar , James Beck , Eugene Gluzberg , Jonathan Munson , Jeremy Sussman , Deborra Zukowski, Challenges: an application model for pervasive computing, Proceedings of the sixth annual international conference on Mobile computing and networking, p.266-274, August 06-11, 2000, Boston, Massachusetts, United States

Michael Samulowitz , Florian Michahelles , Claudia Linnhoff-Popien, Adaptive interaction for enabling pervasive services, Second ACM international workshop on Data engineering for wireless and mobile access, p.20-26, May 2001, Santa Barbara, California, United States

Robert Grimm , Tom Anderson , Brian Bershad , David Wetherall, A system architecture for pervasive computing, Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system, September 17-20, 2000, Kolding, Denmark

↑ INDEX TERMS

Primary Classification:

C. Computer Systems Organization

↳ C.2 COMPUTER-COMMUNICATION NETWORKS

Additional Classification:

C. Computer Systems Organization

General Terms:

Design, Performance, Theory

↑ Peer to Peer - Readers of this Article have also read:

Editorial pointers

Communications of the ACM 44, 9

Diane Crawford

News track

Communications of the ACM 44, 9

Robert Fox

Forum

Communications of the ACM 44, 9

Diane Crawford

Exploiting space and location as a design framework for interactive mobile systems

ACM Transactions on Computer-Human Interaction (TOCHI) 7, 3

Alan Dix , Tom Rodden , Nigel Davies , Jonathan Trevor , Adrian Friday , Kevin Palfreyman

New Products

Linux Journal 1996, 27es

CORPORATE Linux Journal Staff

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2003 ACM, Inc.

Next Century Challenges: Data-Centric Networking for Invisible Computing

The Portolano Project at the University of Washington*

Mike Esler, Jeffrey Hightower, Tom Anderson, and Gaetano Borriello

Department of Computer Science and Engineering
University of Washington, Seattle

{esler, jeffro, tom, gaetano}@cs.washington.edu

Abstract

Computing and telecommunications are maturing, and the next century promises a shift away from technology-driven general-purpose devices. Instead, we will focus on the needs of consumers: easy-to-use, low-maintenance, portable, ubiquitous, and ultra-reliable task-specific devices. Such devices, although not as limited by computational speed or communication bandwidth, will instead be constrained by new limits on size, form-factor, and power consumption. Data that they generate will need to be injected into the Internet and find its way to the services to which the user has subscribed. This is not simply a problem of ad-hoc networking, but one that requires re-thinking our basic assumptions regarding network transactions and challenges us to develop entirely new models for distributed services. Network topologies will be intermittent and services will have to be discovered independently of user guidance. In fact, data transfers from user interfaces to services and back, will need to become invisible to the user and guided by the task rather than explicit commands. This paper outlines a vision of this future and identifies research problems that will require our attention in the areas of user interfaces, distributed services, and networking infrastructure.

1 Introduction

As computing and communication technology advances, we are challenged to build applications where the user interface is not *on* a computer, but *is* the computer; where users do not *connect* to a network, but have their data *travel* on the network; where users are not *commanding* operations to be performed, but agents are operating autonomously on their *intentions*. This will be a major aspect of the coming transition of computing into a consumer mass-market for a wide range of interconnected task-specific devices motivated by function and convenience rather than by technological prowess.

*Portolano charts are the seacoast charts created by Portuguese sailors of the 14th and 15th centuries. They were fundamental to the Age of Discovery initiated by Prince Henry the Navigator that led to the European discovery of the African coast and the New World.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Mobicom '99 Seattle Washington USA
Copyright ACM 1999 1-58113-142-9/99/08...\$5.00

A simple analogy inspired by Donald Norman illustrates our vision [22]. Today's desktops and palmtops are multi-purpose tools – electronic Swiss Army knives. But how many of us would use a Swiss Army knife for preparing a dinner at home? It may be fine on a camping trip, but impractical for more routine activities where efficiency and quality are more highly valued. Instead, we use specialized tools such as paring knives, tongs, stirring implements, and a variety of food processing appliances. Moreover, we have a very easy time borrowing a carving knife from a neighbor or using one in a kitchen other than our own.

This is not the case for computers today. They are too complex because they try to be all things to all people. Although we may encounter them everywhere, we cannot easily use them for our purposes; they often operate in an unfamiliar way, attempt to do too much, or use services we may not (or should not) be able to access. Consumer computing devices should be just as easy and flexible to use as kitchen utensils. The focus needs to shift from the innards of devices to the uses and services that make sense in our daily lives.

Such a conceptual shift becomes possible as technology ceases to be a limiting factor. We can expect in the very near future to pay \$10 for a GB of RAM, a GFLOP/sec of computation, a megapixel of display area, 1Mbs of wireless communication, or 100GB of disk space. This will enable us to take computing capability for granted in everyday devices. New computing capabilities will also be common-place: speech and handwriting recognition are rapidly improving and disseminating, vision-based gesture recognition is coming out of the laboratory, and wireless technologies are proliferating at ever increasing bandwidth.

But before these changes radically alter the consumer landscape, crucial research must be carried out in at least three areas:

1. **User Interfaces:** New modes of interaction such as user movement, proximity of devices, and embodied information presentation will augment the keyboard, pen, audio and video interfaces we see today. The challenge is maintaining task-oriented consistency across physical devices while managing the multiple interfaces in a coherent manner. Also, the focus must shift to user intent and expectation and away from the execution of explicit commands. Data gathered from a variety of location and motion sensors, identification tags, and on-line services, will augment or replace many user directives common today.
2. **Distributed Services:** Rather than abstract capabilities and specific infrastructure items, emphasis must

be placed on applications to which users can easily relate. Different user interfaces, appropriate to their contexts, should be able to interact with the same services. A scheduling service, for example, should be uniformly available via home display, PDA, auto PC, and a phone with voice recognition and synthesis. These services must be more openly organized - into extensible horizontal layers rather than the vertically integrated monolithic services of today - to facilitate service deployment and consumer choice.

3. **Infrastructure:** The networking fabric must provide robust data transfer with replication and discovery as well as the ability to marshal computing resources at internal network nodes. Users must be able to count on their data arriving where it needs to go without their direct intervention. Thus, the network must be data-centric; transmission, routing, authentication, and resource reservation should be handled independently of the location of the user who injected the data. Intermittent wireless connectivity and transmission cost optimizations are examples of decisions for which the user should simply provide guidance once rather than for each use.

The remainder of the paper revolves around these three areas. In Section 2, we elaborate on our vision of mobile computing in the next century with an example that illustrates our ideas. Section 3 describes in detail the specific obstacles to realizing our vision as well as potential partial solutions derived from ongoing research. Finally, we conclude in Section 4 with recommendations for the mobile computing community and outline our own research efforts.

2 The Vision

For too long our computing tasks have been driven by infrastructure and technology. Instead, tasks should be accomplished easily, ubiquitously, and worry-free. The landscape of the 21st century we envision is one where computing devices will be highly specialized to particular tasks, will be ubiquitous consumer items, and their user interfaces will be invisible to all but the most sophisticated users. Users will not be concerned with such esoteric issues as file formats, configurations, and connectivity. They will expect tools to match tasks and not be dependent on ownership or access rights. They will exploit a vibrant, robust, and highly differentiated market of information services to get their objectives accomplished. Moreover, they will trust the communication infrastructure to safeguard their data and deliver it to the services to which they have subscribed or requested.

This terrain is illustrated by the following scenario:

Alice begins the day with a cup of coffee and her personalized newspaper. When her carpool arrives, she switches to reading the news on her handheld display, where she notices an advertisement for a new 3-D digital camera. It looks like something that would interest her shutterbug-friend Bob, so Alice asks her address book to place the call.

Bob's home entertainment system softens the volume of his custom music file as his phone rings. Alice begins telling Bob about the camera, and forwards him a copy of the advertisement which pops up on his home display. Bob is sold on the

product, and after hanging up with her, he asks his electronic shopping agent to check his favorite photography stores for the lowest price and make the purchase.

When the camera arrives, Bob snaps some photos of his neighbor's collection of antique Portuguese navigation instruments. After reviewing the photo album generated automatically by a web-based service, Bob directs a copy of his favorite image to the art display in his foyer. He also sends a pointer to the photo album to Alice and instructs his scheduling agent to set up a lunch date so that he can thank her for the suggestion.

In section three, we will break apart this scenario into its constituent parts and highlight the major issues that it raises, but first, let us briefly speculate about what sort of environment would support this sequence of events.

Alice loses her news connection as she exits her apartment, but it is automatically reestablished via a wide area wireless connection when she turns the next page in her (now PDA-based) personal newspaper. When Bob decides to purchase his new camera, his voice-print and his biometric identification earring authenticate the transaction. His digital camera comes equipped with a short-range radio transceiver as well as a removable cartridge. The datawatch he wears can communicate with the camera as well. When he snaps a picture, the photo data is communicated to the watch where it is replicated and held until a connection can be made to the rest of the world.

While the camera may remain isolated from the network (in the trunk of his car, for example) Bob will eventually enter his home or bring his watch close to his mobile network/telephone terminal. At this point, the phone or, alternatively, the home network portal, can upload the data from the wrist-watch and send it along on the next step of its journey. The pictures find their way through various proxies to the photo album service (As a shutterbug, Bob decided to purchase the album service instead of paying on a per use basis). The costs for all the communications listed above are borne by Bob. When the camera takes the snapshot it included his personal ID (from the earring with biometric safeguards) in the data packets. Lower-cost options are always considered. For example, if Bob is near his home (and likely headed there), the phone will not place a call to transfer the data, but rather defer the transfer to happen at home where a cheaper Internet connection is available.

3 Realizing The Vision

Why is such a storyline not possible now? Each of the areas identified in the introduction (user interfaces, distributed services, and infrastructure) has numerous obstacles which currently prevent the vision from being realized. We will now consider each of these areas in some detail.

3.1 User Interfaces

User interface (UI) issues are of major importance in our vision. Alice and Bob are surrounded by computing appliances, but the interactions occur smoothly and easily. While there are certainly many issues in UI evolution raised by our vision, in this paper we consider two that we feel are especially important.

3.1.1 Multiple Interfaces

The first challenge is handling multiple access points to distributed services. For example, how do we present Alice with a usable newspaper as she leaves her home display and switches to her PDA to ride with her carpool? How do we provide the scheduling service to a nomadic user like Bob as he potentially wanders from home to office to car? He should be able to complete a task such as scheduling the lunch date with Alice as naturally and easily from his car phone or as he can using his home display terminal. His interface should be intuitive on each physical device and yet still clearly derived from the same scheduler service.

Let us first step back for a moment. In the WIMP (Window, Icon, Menu, Pointing) arena, the notion of separating interface issues from semantics is well studied. Window managers, in all their flavors, are quite good at abstracting and managing screen real estate. Also HTML, at least in its intended form, is excellent for specifying content and leaving presentation to the discretion of the clients. Similar techniques might therefore be employed as clients "go mobile" and use new and novel interactions methods such as vocal, handwritten, video, gestural, or olfactory interfaces. One possible quick solution is to force the WIMP approach on mobile devices with the advantage that familiar desktop applications can migrate directly to the mobile devices. While this migration seems appealing, it is flawed since the new constraints and benefits levied by mobile devices are ignored. A better solution is a method to allow mobile clients to discover the semantics of any service's UI and present an interface suited to the client's size, shape, abilities, or resource limitations.

Research toward this goal is already underway. A first attempt was Interface Description Languages (IDLs) as described by Hodes et al [16]. IDLs describe abstract UI semantics via a hierarchical set of types. More recently, IDLs have been superseded by a scheme built on top of the extensible markup language (XML) [15]. XML is a format for using SGML, the Standard Generalized Markup Language (of which HTML is a subset), for document interchange. XML is not a language as much as it is a standard that permits the creation of custom markup syntax to suit the needs of a particular document format [4]. Another interesting research effort is the VoxML markup language from Motorola. VoxML allows the integration of speech interfaces for interaction with web content through simulated dialogues [21].

Markup languages are undoubtedly an enabling technology – the web's rapid growth can largely be attributed to the ease and simplicity they afford to publishing. But as good as XML is at describing the semantics and content of a document, it will never be enough. The abundance of HTML extensions and plugins illustrate that application designers want more control over the precise "look and feel" layout of the interfaces to their systems. There may be an initial embrace of XML-based markup as in the heady days of HTML, but there is no reason to believe that the desire for more precise layout control will not resurface in a mobile multi-interface environment once the novelty of newly enabled technologies such as speech browsing has passed. The challenge is to provide a robust UI architecture to balance the users and developers' needs with the continual growth of the network fabric.

This line of argument suggests a need for advanced development environments that allow developers to create new user interfaces (on specific devices) while re-using much of the code of the back-end of the application (the network communication code and the ties to network services).

3.1.2 Invisible Interfaces

Another interface issue is the movement away from explicit commands and toward interfaces that implicitly take their direction from people's behavior. The challenge is to make the UI fit so well into the environment that it becomes invisible inasmuch as the user is aware she is interacting with a computing device. Such design is enabled by involving users in the design process. This idea is the thrust of much recent work in the HCI community including that of Fishkin et.al.[10] and Weiser [39].

To achieve this invisible design, we must first conduct user studies, create prototypes of the new mobile world, and deploy applications and services that a wide range of users will find useful. This process is apt to be a multidisciplinary effort involving talent from new fields traditionally not directly related to computing such as anthropology, human factors, and ethnography. Along the way we will develop the networking protocols, distributed services, location sensing infrastructure, and a collection of devices that demonstrate the effectiveness and utility of our approach.

One potential dividend of invisible computing research is context aware computing (CAC). CAC attempts to coalesce knowledge of the user's task, emotions, location, and attention with other available data such as the time and knowledge about other users. The CAC field is in its infancy, but already groups are exploring this design space with projects such as Georgia Tech's CyberDesk [1] and the spatial location work at AT&T Laboratories Cambridge [37] and Xerox PARC [36] [35]. The likely result is that the fusion of data from a variety of sensors and databases will be crucial to inferring intention. A major issue will be in how to get the required information in a timely an efficient manner.

3.2 Distributed Services

The core of our research should be in distributed services. Instead of computing infrastructure with abstract capabilities, we must instead provide the user with services to which they can easily relate. For example, when Bob creates his new photo album, he may call on several different services to operate on his photos. One of these stores the photos on a reliable server in his home or a rental site. Web pages are then prepared by the service and stored as part of the subscription arrangement. The request to forward his favorite photo to the foyer display is handled by the photo album service. Because satisfying the request may involve the purchase of rights to images, the photo album service checks with Bob's financial management service to see if he is on budget and the purchase meets his expected usage profile. Once the purchase is made, the agent may then negotiate with a home decorator service to determine in which situations the image should be displayed in the foyer (for example, especially when Alice pays him a visit). Bob does not concern himself with the technical specifications of the photo server nor did he muddle through file format conversions. Such issues are the concern of the service provider Bob employs. To Bob, the infrastructure is simply a means to an end.

3.2.1 Horizontal Integration

The current infrastructure-centric focus has led to system architectures that are vertically integrated, not horizontally layered. By vertical we mean systems that they attempt to provide entire solutions to a problem. Traditionally, these

suffer from high-cost and inflexibility (e.g. the inability to get information to and from users who do not subscribe to the same service). Although administration and regulation can be centralized, vertical systems often make it difficult or even impossible for a user to obtain exactly the subset of services he requires (the "take it or leave it" dilemma). Furthermore, vertical systems tend to stifle competition and make it difficult to quickly deploy new or alternative services.

We argue that horizontal layering is much more appropriate for the mobile networks of the future. If Bob desires to migrate to a new photo album service or even use the same service with different mass storage arrangements or more exciting graphic design it should be possible for him to easily do so.

3.2.2 Agent Technology

Many of the tasks that Alice and Bob performed in the scenario require the unintentional use of a variety of distributed services. When Bob scheduled lunch with Alice, he did not explicitly (intentionally) access Alice's calendar and find an available date. A scheduling *agent* performed this task on his behalf - taking into account both Alice and Bob's schedules, eating habits and preferences, and convenient travel and meeting times. Yet another agent assisted Bob in publishing a photo album by making layout and presentation decisions based on his preferences.

The term *agent* is difficult to define. Nwana describes an agent as a "component of software and/or hardware which is capable of acting exactly in order to accomplish tasks on behalf of its user." [23] The technology and protocols used to implement agents are becoming better understood [9] [8], but how they can be applied to mobile applications in environments with widely distributed data sources and intermittent connectivity is research that must be further explored. Of particular importance will be an active networking structure allowing the flexible integration of applets and servlets.

3.2.3 Service Deployment

Smooth integration of new services is essential. If Bob were to purchase a new photo service such as face recognition for automatic portrait labeling, its installation should occur seamlessly. The service should be able to discover Bob's resources (rented or owned) such as storage, processing cycles, etc. necessary for its operation. Similarly, a new hardware component must be able to setup itself and its connection without the explicit involvement of a network provider or other such middlemen. In the current model, significant configuration is required for new devices such as wireless phones.

Discovering available resources, both in general and at any specific point in time, is crucial to effective service deployment and should be an automatic process from the user's perspective. Resource discovery is an infrastructure issue and thus will be addressed in some detail in section 3.3.1

Deploying services effectively also necessitates new distribution and maintenance models. In current systems, users often feel overwhelmed from perpetually having to upgrade their systems with new enhancements, bug fixes, and security patches. Although the user must certainly be kept in the loop about major issues, the day to day maintenance chores should be the responsibility of the service or subscription provider. This model implies the creation of an architecture supporting dynamic upgrading and hot-swapping of system

components. Indeed, many vendors are already addressing this issue. For example, web browsers and multimedia playback tools often come equipped with these autoupdate abilities in order to simplify supporting new codecs and data formats. The challenge will be to implement similar techniques in the mobile domain while spanning heterogeneous hardware and connectivity situations.

3.3 Infrastructure

While task-oriented applications will be the motivation of our research, we expect to encounter many interesting infrastructure challenges in their implementation. The nature of these challenges will be applying existing technologies to the mobile and invisible domains.

3.3.1 Resource Discovery

In our scenario, Alice and Bob's devices were able to discover the network services available and communicate with each other without the end-user's assistance. When Alice asked to call Bob, her address book discovered the phone service available in the car, downloaded an interface to that phone, and invoked a remote method using the interface. Similarly, Bob's phone was able to discover the stereo system, retrieve its interface, and pause the music.

Resource discovery is the subject of numerous research efforts including the RDP [27] and SLP [25] protocols, Berkeley's Service Discovery Service [5], Sun Microsystems' JavaSpaces and Jini [34], T-Spaces from IBM [29], Universal Plug and Play from Microsoft [28], and the HAVi consumer electronics consortium [14]. Each takes a slightly different approach based on the application domain they were intended for. None were designed specifically with mobile networks in mind, so a host of questions should be re-considered in this new context: Should resource discovery depend on a local service database, or are ARP-style requests more appropriate? Should a lookup service provide clients with a resource name, or an object written in Java or object-TCL? How powerful should the lookup query model be, and how should it be implemented: as Java objects, XML, or something else? In addition to answering these questions, it is important that any discovery systems we implement be self-managing since both clients and resources (including the lookup registry) are likely to change as devices are disconnected and reconnected.

Due to intermittent connections and ad hoc networks, data may have to find services on its own without the assistance of the application that injected them into the network. This necessitates the ability of the networks to execute code in the data packet. This code can call on discovery functions provided on major nodes or, if it finds itself on minor nodes that only route, select the best route to follow to get to those services. To guarantee data safety, this will also require controlled replication of data packets and finite lifetimes. Acknowledgments from the services that receive the data packets back to the original generators of the data are also problematic as the source may be disconnected or may have moved to a new location. Thus, replies also need to be able to call upon distributed location services (that may in turn call upon schedule and profile services) to help them find their routes or cache data in anticipation of a future connection.

While it is important for devices to be able to discover services, they should only be able to use those services for which they have permission. Bob's camera, for example, did not publish the photos in his neighbor's album. Likewise,

Alice's address book can only place calls from phones that she is allowed to use. Kerberos [30] is currently a popular system for authenticating network requests, but its statically configured centralized servers make it a poor choice for mobile networks. The IPsec protocol [17] is a newer scheme for private data transfer built on public-key infrastructure, which may be better suited to the needs of mobile computing. But IPsec does not solve the problems of privacy and authentication central to our vision. The SDS protocol also provides security, in the form of DSA signatures, and RSA encrypted broadcasts. Significant challenges remain, however: What does a device do if the network authentication server is not available? How do you revoke certificates of devices that are only intermittently connected? For example, can Alice's daughter have a private video diary that is kept secret from others in her family as well as protected from outside access?

3.3.2 Data-Centric Networking

Another key issue is the need for data-centric networks. Active data bundles should be able to marshal (and pay for) the resources they need to make progress in the network. Bob's camera, for example, is capable of transferring photos to several locations without the aid of cables or a base station. Its RF transmitter can communicate with nearby devices including Bob's wristwatch, cellular phone, and appliances. Data moves from device to device until it reaches the service it is intended for. Though the ideas of ad-hoc networking are valuable, we need to re-think our basic assumptions about network operations and construct a data-centric network architecture as a means for distilling, naming, and locating the data objects that travel within the network.

When Alice leaves home, her personalized newspaper is still available to her. It is certain that the network characteristics of her home and car are very different, so the quality of the newspaper adapts. A high-bandwidth service may be essentially free at home, but costly to use elsewhere. Network infrastructure must be able to inform devices about the network they are using, as well as be able to provide admission control. Unlike the Internet, Bluetooth [32], HomeRF [11], USB [33], and IEEE 1394 [18] networks all reserve bandwidth for synchronous data channels. In these cases, the network infrastructure should be able to offer guarantees about the quality of service (QoS) to those users willing to pay a premium, provided the network has sufficient free capacity. Protocols such as RSVP [41] and QEX [6] have only laid the initial groundwork for effective QoS management in mobile applications. Those services which do not have specific QoS requirements may still benefit from knowledge about the current network state.

A data-centric network must also be able to manage ubiquitous persistent storage. Bob's photo album may reside on a specialized storage device in his home that his camera, art display, and friends can all access simultaneously or, alternatively, Bob's photos may reside in several different locations - redundantly or in different forms. Sun Microsystems's NFS [20] offers transparent and authenticated access to a global set of files residing on a central server. Unfortunately, this system requires static configuration and has a single point of failure, making it undesirable for mobile applications. Reliability and availability can be increased by using a distributed file system such as the Open Software Foundation's DFS [24]. At the same time, DFS provides users with access security via Kerberos, and a *uniform name space* for accessing files. The Coda filesystem [31] is

a descendant of AFS that is designed specifically for mobile clients. The Bayou project [7] provides mobile clients access to data by using a distributed database approach. Using these projects as starting points, a combination of storage services will need to be designed and integrated into mobile environments. What is really needed to make our vision a reality is ubiquitous storage made available to distributed applets running across an ad hoc network. We still have much to study about the consistency semantics in this type of environment in the presence of failures and intermittent connections. None of the existing systems are adequate for these purposes.

3.3.3 Distributed Computing

The infrastructure and technology of distributed computing also plays an important role in building the services illustrated in the scenario. When Bob sent his photos to the photo album service, they needed to be in a specific compression format. Rather than waste bandwidth and process the photos when they arrive at the service, the camera downloads software to run locally. How did the camera know what methods this object exports? Bob's phone cannot store interfaces for every device in his home, so it must request the correct interface whenever it needs to execute a remote method. How is this accomplished?

Several emerging solutions to this problem, including Jini, Liquid Software [13], and the Active Networks Toolkit [40] are based on the Java language. In these models, bytecode is downloaded and executed on the client. Using the Java RMI, clients can then use the services of other devices. Other solutions are based on CORBA [12] and Microsoft's COM [19]. Both allow clients to execute code located elsewhere, and provide mechanisms that allow clients to discover an object's interface at runtime. Unlike the Java-based solutions, CORBA and COM place restrictions on the interface rather than the implementation language. The challenge will be in coming to a consensus and developing an open standard that incorporates the positive aspects of each.

In section 3.1.1 we suggested that application development environments should automate multiple user interface creation, but ideally development environments could also be built to synthesize all the detailed communication and coordination code and optimize it for the needs of the application. These needs could include restoration of hard and soft state after network partitions and transfer of data in appropriate bundles for service discovery, authentication, and data migration. As it is likely that the code will need to targeted to very different architectures and operating systems, mitigating the development burden - synthesizing communication and coordination code and deploying it automatically onto a distributed fabric - will be crucial to ensuring that applications are written by as large a segment of the population as possible. This automatic code synthesis is similar in spirit to CAD tools for distributed embedded systems such as Chinook [2].

3.3.4 Intermittent Connectivity

In order to achieve invisible, trouble-free connections and disconnections from networks, mobility must be built into our protocols. Indeed, we find it likely that intermittent connectivity will be the norm for the foreseeable future due to power, cost, bandwidth, latency, and congestion limitations. By disconnecting during idle periods, devices will consume less power, and lengthen the time between recharging batteries. Not surprisingly, new mobile protocols are

already appearing for the intermittent connection environment. Bluetooth [32] and HomeRF [11] are standards for small area RF networks in which devices can join and leave ad-hoc networks of devices as necessary. While this technology will allow mobile devices to join new local networks to take advantage of local resources, it does not address more complex issues such as hand-off and signal strength analysis found in cellular protocols [38]. Mobile devices also need to be able to ascertain information about the networks that they join. Some solutions leverage off of the DHCP and DNS protocols [3], but their use is limited to IP networks. Other important issues, including IP tunneling and mobile host registration, are contained in the mobile IP specification [26].

Each technology is important to the future of mobile computing, but no single protocol is completely satisfactory. The cellular model works well, but is it appropriate for the desired services and applications? More likely we will need a range of wireless technologies with different transmission ranges and power requirements to support devices like key-chains and earrings that function indefinitely without recharging but have very limited range to more traditional PDAs that need to connect to the Internet and can be more easily recharged. Can the existing infrastructure support a large increase in the number of devices connected? RF is promising, but it suffers from a limited bandwidth per volume defined by its range - another reason for using a wide range of wireless technologies and overlaying their ranges. Many devices will want to operate in multiple overlays and act as routers for others. Irrespective of the medium, security must be integrated into the protocols to satisfy application requirements.

4 Recommendations

We have presented a vision that we believe will require fundamental new work in several disciplines if it is to become a reality. Key features of our vision include:

- multiple user interfaces that rely upon user intent,
- horizontally-layered network-based services, and
- novel routing and active networks infrastructure.

We strongly believe that any exploration should begin with user studies, analysis of current practices, and the building of complete usable systems. It is imperative that we shift from a technology to an application focus. We must determine how our prototypes will meet the challenges of user-centric tasks, and how these motivate the new devices and services.

Simple applications motivated by needs of the office and home environments can serve as a starting point. An example that demonstrates the extreme interwoven nature of our vision is a calendar application that ties together a variety of input devices (including PDAs, desktops, telephones, location sensors, and wall-mounted displays) with a variety of actuators (including alarms, e-mail notification, and automobile navigation systems) with a variety of services (including individual and group diaries, negotiation agents for scheduling meetings, and photo album services). For example, data fusion should make it possible for the calendar service to adjust the morning wake-up alarm based on traffic conditions or the knowledge that the car is almost out of gas. Building these devices will expose interesting issues in all three of the thrust areas.

We are beginning our research work by looking at both applications and infrastructure issues in parallel. In the applications area, we are experimenting with the practical uses of in-building location sensing based on RF tags. Our hope is not to track people but objects of a wide range of sizes - from folders to books to appliances - via triangulation over time with connection to physical layout and the expected uses of objects and locations. Another application area involves the dissemination of small embedded web servers (2 sq.in. boards with 10-baseT Ethernet and serial connections) into the instructional spaces in and around our building and campus acting as portals to our department's network. On this infrastructure, we initially plan to deploy a simple messaging service and architecture robust enough to support future applications such as distributed calendars and scheduling.

In the infrastructure area, we are evaluating different approaches to service discovery and more specifically the architectures for proxies to handle computationally-limited mobile devices. In networking, we are focusing on creating a general-purpose API for collecting data from varied sensors, operating on the composite data, and controlling actuators with the results of those computations. Our hope is to develop an open-source library to enable others to work in this space.

We expect the constraints that arise due to:

- intermittent connectivity,
- power consumption,
- application development and deployment,
- service architectures and discovery, and
- active networking

to dominate our research agenda in making invisible computing a reality.

5 Acknowledgments

We acknowledge stimulating collaborations, discussions, and comments from Roy Want of Xerox PARC, David Wetherall of the University of Washington, the DARPA Expeditions program and David Tennenhouse director of DARPA/ITO, and the students of CS-590ES - particularly Kurt Partridge.

References

- [1] G. Abowd and C. Atkeson. Future computing environments: Cyberdesk. Technical report, Georgia Institute of Technology, 1998. <http://www.cc.gatech.edu/fce/cyberdesk/>.
- [2] P. Chou, R. Ortega, K. Hines, K. Partridge, and G. Borriello. ipChinook: An integrated IP-based design framework for distributed embedded systems. In *DAC-99*, 1999.
- [3] D. Comer. *Internetworking with TCP/IP*, volume 1. Prentice Hall, Upper Saddle River, NJ, third edition, 1995.
- [4] World Wide Web Consortium. Extensible markup language (XML), 1998. <http://www.w3.org/XML/>.

- [5] S. Czerwinski, T. Hodes, B. Zhao, and A. Joseph. The service discovery service: Searching for computation power, 1999. <http://www.cs.berkeley.edu/czerwin/sds-project.html>.
- [6] N. Davies, A. Friday, G. S. Blair, and K. Cheverst. Distributed systems support for adaptive mobile applications. *Mobile Networks and Applications*, 1(4):399–408, 1996.
- [7] W. K. Edwards, E. D. Mynatt, K. Petersen, M. J. Spreitzer, D. B. Terry, and M. M. Theimer. Designing and implementing asynchronous collaborative applications with bayou. In *UIST '97*, Banff, Alberta, Canada, 1997. ACM.
- [8] O. Etzioni, S. Hanks, and D. Weld et al. Internet softbot research, 1998. <http://www.cs.washington.edu/research/projects/softbots/www/softbots.html>.
- [9] Tim Finin. UMBC agent web, 1998. <http://www.cs.umbc.edu/agents/technology/>.
- [10] K. P. Fishkin, T. P. Moran, and B. L. Harrison. Embodied user interfaces: Towards invisible user interfaces. Technical report, Xerox Palo Alto Research Center, Palo Alto, CA, 1998.
- [11] HomeRF Working Group. HomeRF, 1999. <http://www.homerf.org/>.
- [12] Object Management Group. Corba news, 1998. <http://www.omg.org/corba/>.
- [13] J. Hartman, U. Manber, L. Peterson, and T. Proebsting. Liquid software: A new paradigm for networked systems. Technical report, University of Arizona, Tucson, AZ, 1996.
- [14] The HAVi specification. Technical report, Home Audio/Visual Interoperability Consortium, 1998. <http://www.havi.org/>.
- [15] T. D. Hodes and R. H. Katz. Enabling “smart spaces:” entity description and user interface generation for a heterogeneous component-based distributed system. In *DARPA/NIST Smart Spaces Workshop*, Gaithersburg, Maryland, 1998. DARPA/NIST.
- [16] T. D. Hodes, R. H. Katz, E. Servan-Sreiber, and L. Rowe. Composable ad-hoc mobile services for universal interaction. In *Mobicom '97*, pages 1–12, New York, NY, 1997. ACM.
- [17] S. Kent and R. Atkinson. Security architecture for the internet protocol, 1998. RFC 2401.
- [18] A. J. Kunzman and A. T. Wetzel. 1394 high performance serial bus: The digital interface for ATV. *IEEE Transactions on Consumer Electronics*, 14(13):893–900, 1995.
- [19] Microsoft. Microsoft COM technologies, 1998. <http://www.microsoft.com/com/>.
- [20] Sun Microsystems. The NFS distributed file system. Technical report, Palo Alto, CA, 1995.
- [21] Motorola. Motorola VoxML, 1998. <http://voxml.motorola.com/>.
- [22] D. Norman. *The Invisible Computer*. MIT Press, Cambridge, MA, 1998.
- [23] H. S. Nwana. Software agents: an overview. *Knowledge Engineering Review*, 1(3):205–244, 1996.
- [24] File systems in a distributed computing environment. Technical report, Open Software Foundation, 1991.
- [25] C. Perkins. SLP white paper. Technical report, Sun Microsystems, 1998. <http://playground.sun.com/srvloc/>.
- [26] C. E. Perkins. IP mobility support, 1996. RFC 2002.
- [27] C. E. Perkins and H. Harjono. Resource discovery protocol for mobile computing. *Mobile Networks and Applications*, 1(4):447–455, 1996.
- [28] Universal Plug and Play Forum. Universal plug and play. <http://www.upnp.org/>.
- [29] IBM Research. IBM T-Spaces project. <http://www.almaden.ibm.com/cs/TSpaces/>.
- [30] Frequently asked questions about today's cryptography. Technical report, RSA Data Security, Inc., San Mateo, CA, 1998.
- [31] Carnegie Mellon University School of Computer Science. Coda file system, 1999. <http://www.coda.cs.cmu.edu/>.
- [32] Bluetooth SIG. Bluetooth technology, 1999. <http://www.bluetooth.com/technology/>.
- [33] Universal serial bus specification, revision 1.1. Technical report, Compaq, Intel, Microsoft, and NEC, 1998. <http://www.usb.org/developers/data/usb11.pdf>.
- [34] J. Waldo. Jini architecture overview. Technical report, Sun Microsystems, Inc., Palo Alto, CA, 1998.
- [35] R. Want, K. Fishkin, B. Harrison, and A. Gujar. Bridging real and virtual worlds with electronic tags. Technical report, Xerox Palo Alto Research Center, Palo Alto, CA, 1999. Also in CHI 99 (supporting video available).
- [36] R. Want, B. Schilit, N. Adams, R. Gold, D. Goldberg, K. Petersen, J. Ellis, and M. Weiser. The PARCTAB ubiquitous computing experiment. In T. Imielinski, editor, *Mobile Computing*, chapter 2, pages 45–101. Kluwer Publishing, February 1997.
- [37] A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. *IEEE Personal Communications*, 4(5):42–47, 1997.
- [38] Wireless data primer. Technical report, Wireless Data Forum, Washington, D.C., 1998.
- [39] M. Weiser. Open house. *Review*, 2, 1996.
- [40] D. J. Wetherall, J. V. Guttag, and D. L. Tennenhouse. ANTS: A toolkit for building and dynamically deploying network protocols. In *OPENARCH '98*, San Francisco, CA, 1998. IEEE.
- [41] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE-Network*, 7(5):8–18, 1993.

[> home](#) [> about](#) [> feedback](#) [> login](#)

US Patent & Trademark Office

Citation

StandardView [>archive](#)Volume 6 , Issue 3 (September 1998) [>toc](#)

XML: not a silver bullet, but a great pipe wrench

Authors

Tommie Usdin Mulberry Technologies, Inc., Rockville, MD

Tony Graham Mulberry Technologies, Inc., Rockville, MD




Publisher

ACM Press New York, NY, USA

Pages: 125 - 132 Periodical-Issue-Article

Year of Publication: 1998

ISSN:1067-9936

 <http://doi.acm.org/10.1145/324042.324049> (Use this link to Bookmark this page)[> full text](#) [> index terms](#) [> review](#) [> peer to peer](#)[> Discuss](#)[> Similar](#)[> Review this Article](#) [Save to Binder](#)[> BibTex Format](#)[↑ FULL TEXT:](#)  [Access Rules](#) [pdf 87 KB](#)[↑ INDEX TERMS](#)**Primary Classification:**

I. Computing Methodologies

↳ I.7 DOCUMENT AND TEXT PROCESSING

↳ I.7.2 Document Preparation

↳ Nouns: XML

General Terms:

Languages

[↑ REVIEW](#)

"Edouard J. Desautels"

This eight-page paper does a good job of explaining what XML is and is not, and what it can be used for. The relationship between XML and HTML is described, as is its connection to SGML. The paper provides several Web URLs for sites working on XML-related activities. The section on shared or community tag sets lists specific works in progress, ranging from an astronomical markup language to GEDCOM, for genealogical data. The authors briefly describe some nine e-commerce initiatives based on XML. Readers who want an executive overview of XML and insight on why it may be important would do well to track down this article. *Online Computing Reviews Service*

↑ Peer to Peer - Readers of this Article have also read:

MBONE: the multicast backbone

Communications of the ACM 37, 8

Hans Eriksson

Synthesis of complex dynamic character motion from simple animations

ACM Transactions on Graphics (TOG) 21, 3

C. Karen Liu , Zoran Popović

Geometry images

ACM Transactions on Graphics (TOG) 21, 3

Xianfeng Gu , Steven J. Gortler , Hugues Hoppe

News track

Communications of the ACM 44, 9

Robert Fox

Editorial pointers

Communications of the ACM 44, 9

Diane Crawford

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2003 ACM, Inc.



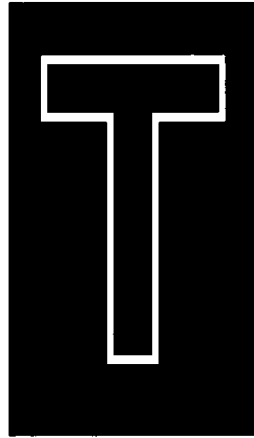
XML: Not a Silver Bullet, But a Great Pipe Wrench

*Tommie Usdin and
Tony Graham*

MULBERRY TECHNOLOGIES, INC., ROCKVILLE, MD

■ XML (Extensible Markup Language) provides both a standards-based way to identify the information that is of importance in a particular application, and the ability to process information tagged according to highly user-specific requirements with general-purpose software, such as editing tools, composition engines, and electronic browsers. The power of XML comes in part from principles that guide the design of good XML applications: separation of format and presentation information from document markup; consistent and clear text tagging; context-dependent processing; and hierarchical structures. But these alone do not explain the real power of XML, which lies in the ability to create tag sets and markup languages customized to the needs of the particular application. A custom XML tag set allows the user to identify all of the types of information that are needed for search and retrieval, formatting, and tracking. Any type of information your end users may want to find, or not find, can be identified, and expensive distinctions among types of information that are not important to you are not made.

Note, however, these phrases from the preceding paragraph: "way to identify"; "ability to create"; and "can be identified." XML provides a way to do these things, but does not do them. XML should be thought of as a useful tool, but not as a solution to any problem.



here seems to be as much excitement about XML as there has been on any related technology since the Web went public. The hype surrounding XML has created such unreasonable expectations that there are already people trumpeting its failure, primarily because it hasn't become instantly ubiquitous.

XML is being hailed as the future of the Web, the replacement for HTML, the replacement for Java, and the technology that will create precise Web searching. XML will be easier to use than SGML, more powerful than HTML, and will enable secure electronic commerce. XML is the Internet's Silver Bullet—such is the hype.

XML will not leap tall buildings at a single bound, nor will it solve all of the problems of retrieval on the Web. XML will transform the Web in much the same way barbed wire transformed the American West. Barbed wire didn't do anything. But using barbed wire, a lot of people did a lot of hard work and changed the culture from one of open ranges to one of farms and property rights.

XML is an enabling technology; well designed XML can provide a valuable tool in the effort to provide more precise and more powerful searching on the Web. XML will replace HTML in those situations in which HTML is insufficient to meet a need. XML software is easier to build than SGML software and more appropriate for Web environments, but authoring documents in XML is unlikely to be any easier than authoring in SGML. The ease of authoring in both XML and SGML is dependent on how well the document structure meets the author's needs and on how graceful the authoring application is. And interchange of information in XML depends on the development and promulgation of shared XML tag sets.

What is XML?

Technically, XML is a data format. Or, more precisely, XML is a way to define data formats, a few predefined information types, an optional way to communicate details about the data format, and a style of formatting data. For all practical purposes, XML is also a philosophy of data identification and a host of languages and specifications for functions related to data.



Philosophy behind XML

The philosophy and design of XML are based on a few fundamental principles:

- separation of content from format or processing;
- hierarchical data structures;
- embedded tagging; and
- user-definable structures.

Separation of Content from Format

The most important concept behind XML is that the identification of what a piece of information is should be separated from information on how that information should be presented or processed. Further, information should be identified on as abstract a level as possible. That is, if the same piece of information can be identified by:

- What it looks like;
- How it is used in a particular application;
- Its role in this document or information collection; and
- The nature of the information itself,

it is far more useful to identify the nature of the information itself. Knowing that a phrase is in italic is useful; knowing that it is the title of a subsection of a paper is more useful; and knowing that it is a genus and species name is potentially more useful still.

By identifying what the information is and separately identifying how it should be displayed or processed, multiple uses are possible. For example, the heading above this section should be identified as the heading of a section rather than as text that is formatted in bold and using a specific font. Separately, in a style sheet, there should be information on how headings of sections of this type should be displayed. This separation of content and format means that there can be many different formats and uses of the same content without requiring any change to that content. For example, a large-print or voice-synthesis version of this article could be produced from the same XML file as that from which the print version was produced.

Identification of what the information is also allows information to be used in different ways by different applications and to be processed in ways that were not anticipated at the time the data were created.

This does not mean that format and processing are not defined, nor does it mean that they cannot be defined by the creator of the content. It simply means that they are managed separately. It is likely that there will be many documents created with the assumption that the same style sheet will direct their processing, and that there will be several style sheets used with the same documents at various times or for various uses.

Hierarchical Data Structures

XML assumes that data are hierarchically structured, and that many types of useful and identifiable information contain other useful and identifiable types of

information. The named types of information in XML are called *elements*. XML data are packaged into "document instances" (which most non-XMLers would call documents), which consist of a named "root" element which begins at the beginning of the document, contains the entire document, and ends at the end of the document. It is common for many of the elements that are inside the root element to contain other elements, which may in turn contain yet other elements.

This hierarchical structure is fairly obvious in text documents: a section starts before the heading that starts that section and ends just before the next heading of the same level. A section may contain a heading, some paragraphs, and perhaps subsections that contain headings and paragraphs of their own. This hierarchical structure is very valuable in rearranging documents, or in cutting parts of one document out and inserting them into another; if a user wants to grab the part of a document that addresses a specific topic, s/he will want the heading plus all of the content until the next heading.

Hierarchical structure may be less obvious in non-text documents, but even there it is usually useful to divide the information into a small number of groups, which are then subdivided. For example, a catalog order may consist of information about the customer, information about the items ordered, and information about payment. Each of these groups may contain additional details.

XML document instances are simple tree structures; each instance has one and only one root and is divided into non-intersecting branches. There are no loops in XML document structures.

Embedded Tagging

In XML, *tags* are embedded in the data to identify where each element begins and ends. Tags are words or word-phrases, enclosed in pointy brackets. Readers may be familiar with the tags used in HTML, which identify where headings and paragraphs start, and where bold (or blink) begin and end.

Much of the flexibility of XML comes from the fact that the information elements are identified by tagging embedded in the data. This means that there is no need to specify ahead of time how long any piece of content will be, or in precisely what sequence the data will occur. It means that applications can tolerate a great deal of variation in the data they receive because they will be able to identify the information of interest to them.

In addition to tags that identify the beginning and end of elements in XML, the XML syntax supports *attributes*. Attributes, in the form of pairs of names and values, reside in start tags and provide information about the entire contents of the element.

All XML elements have explicit beginnings and endings, typically using start and end tags, but in the case of empty elements using a hybrid "sole" tag.



User-Definable Structures

XML defines a way to create tags to identify information that is of interest to a user or group of users. XML is not a set of tags. XML assumes that users will create new tags as they create and work with documents, and that software such as browsers will have to display or process the content of these novel tags.

Perhaps the most surprising thing about XML, especially to users of HTML, is that XML neither provides nor recommends tags. It is a fundamental premise of XML that it is not possible to list all of the types of information that any user group may want to identify for interchange or processing. XML provides a mechanism for users—or groups of users—to create their own tags for the information of interest to them.

Many HTML users are shocked by this premise. They want to know how the average user will possibly cope with the requirement that s/he identify his or her own tags. They imagine that chaos will result on the Web when everyone uses their own tags; information retrieval will be even worse than it is today. They demand to know when the XML Committee will publish the starter set of tags. They panic at the notion that in order to browse XML data the rendering engine (browser, print formatter, voice synthesizer, etc.) will need not only the document instance but also guidance on how to render the contents of that document based on the elements and attributes.

General-purpose XML tools will need far more information than HTML browsers have needed; style information must be provided with novel XML elements in order to be rendered. This does not mean that the developer of each XML document must develop a stylesheet, but it does mean that when the author of a document selects a tag set, s/he must either adopt a stylesheet with it or create a new stylesheet, and that when an author creates a new element, it will be necessary to create a style for that element. It is also likely that rendering tools such as browsers will make “educated guesses” about the intended use of an element (is it an inline element, such as an emphasized phrase, or a block element, such as a paragraph?) and provide default rendering based on that analysis.

Community Tag Sets

While XML will not include a set of tags, many communities of XML users will create, or have already created, shared sets of tags. Industry or interest groups will create and share sets of tags, documentation on the use of these tags, style sheets for the display or processing of the tags, and (perhaps) tools for the creation and/or use of documents using the shared tag set.

Some of these XML applications—as the combination of a set of tags, documentation, and associated information is called—are being developed by public groups and industry consortia. Some are likely to be developed and supported by for-profit organizations, such as software vendors. Others are being devel-

oped and supported by non-profit organizations and associations.

The XML Data Format

For full details on the data format defined by XML the reader should, of course, see the XML Recommendation (<http://www.w3.org/TR/REC-xml>). XML will look very familiar to users of the World Wide Web because it looks quite similar to HTML, the language of the Web. This is no coincidence; HTML was inspired by SGML, and XML is an application profile of SGML. SGML is an ISO standard for the markup of text, ISO 8879:1986. Like XML, SGML is a way for users to create markup languages (sets of tags, etc.). HTML varies from being an application of SGML to being a set of tags that sort of look like SGML—that is, the HTML tags can be used in ways that are valid SGML, but often they are not.

Like HTML, XML markup consists of tags and attributes. Matching tags must mark the beginning and end of each element. Attributes, which are embedded in the start tag, may provide additional information about the element.

The Tag Set

HTML is a set of tags that grows slowly, and that for all practical purposes is under the control of the creators of HTML browsers. If a browser can display or process a tag, then that tag can be used, at least in documents intended to be displayed using that browser. Users have the choice to use or not use tags the browsers can accommodate, but users cannot add to the set of tags.

XML puts control of the tag set in the user's hands. Users can create new tags as needed, and it is the obligation of the software that will be processing the data to accommodate the tags created by the users.

Document Type Definitions

XML provides a mechanism to specify what tags will be used in a class (type) of documents, what the permitted relationships are among those tags, what attributes are allowed on each tag, and what the values of those attributes may be. This information is encoded in a Document Type Definition (DTD). DTDs may be used to drive XML-savvy tools that assist authors in creating documents: by leading them through the process of creating sequential information in the correct order; by providing menus of only those elements that are valid in a particular context; and by informing authors when documents are missing required information or otherwise violate the rules for a document type.

The syntax for DTDs specified in XML 1.0 is a subset of SGML's DTD syntax. A DTD can specify what the root element is for a document type, what elements may occur in the document type, what relationships are permitted among these elements, what attributes are allowed on each element, and what values these attributes may take. There is a limited



capability for expressing element relationships and a limited capability to specify attribute content.

There are also several proposals for alternate and richer ways to define/describe the contents of XML documents. Among the features under discussion are the use of XML syntax for defining XML documents, richer constructs for modeling element relationships, the ability to define the contents of elements and to more precisely define the contents of attributes, and to define relationships among the contents of various elements. A work group to address these issues has recently begun to examine them, and it is likely that XML DTD syntax will be supplemented or replaced with a richer data definition language.

Related Specifications

The XML Recommendation is the first member of the XML family to reach W3C recommendation status, but there are several proposed recommendations and W3C Notes related to XML, plus recommendations, notes, and standards from other bodies for applications of XML. These related specifications will provide shared ways to communicate how XML information should behave, be accessed, and be displayed. Their wide adoption will significantly improve the ease with which complex applications can accommodate unknown tags and tag structures.

It is important to note that none of these related specifications addresses what tags users should use for the basic content of their information. However, some do provide semantics for specifying how links (such as hypertext pointers and targets) should be identified and for specification of style information.

XML Specifications

- XML—Extensible Markup Language (<http://www.w3.org/WD/WD-xml>). The base document for the XML family of standards and for all XML applications.
- XLink—XML Linking Language (<http://www.w3.org/TR/1998/WD-xptr-1998303>). Additional functionality for advanced hypertext and hypermedia capability such as links to multiple destinations, bi-directional links, links annotating read-only documents, specification of link behavior, and databases of links. XLink is based on, but is not identical to, the hyperlinking functionality of HyTime, i.e., ISO/IEC10744.

The previous XLL work was split into XLink and XPointer (see below) to separate the linking functionality from the addressing of the objects being linked. See <http://www.w3.org/TR/NOTE-xml-link-principles> for additional information on the design principles of both XLink and XPointer.

- XPointer—XML Pointer Language (<http://www.w3.org/WD/WD-xptr>). Specification of constructs for addressing into the internal structures of XML doc-

uments. This includes absolute locations, such as the root of the document or an element with a specified ID, and locations specified relative to known locations. The relative locators can navigate forwards, backwards, up, and down through the element tree. XPointers can specify spans as well as single points, and can address text regions as well as elements. XPointers are based on, but are not identical to, TEI extended pointers.

- XSL—Extensible Style Language (<http://www.w3.org/TR/WD-xsl>). The language for stylesheets for XML documents. The processing model specifies a transformation stage to construct a “result tree” from the input “source tree”; then the styles are applied to the result tree. The result tree can be completely different from the source tree, and the result tree may use only parts of the source tree, re-use it all, or re-use it multiple times, as well as add generated structure that was not in the source tree.
- XML Namespaces (<http://www.w3.org/TR/1998/WD-xml-names-19980801>). A simple method for qualifying names used in XML documents by associating them with namespaces identified by a URI. Use of namespaces will promote re-use of markup, and will allow composite documents built from fragments from many sources, since the URI will disambiguate element types that would otherwise have the same name. Namespaces will also promote re-use of the software modules that process the markup, since a module can act on element type names within a known namespace and ignore all others.

XML Applications

We anticipate that users will adopt applications such as these for tagging specific types of information, and will integrate these “utility” tag sets into tag sets that describe their information content.

- MathML—Mathematical Markup Language (<http://www.w3.org/TR/REC-MathML>)

An XML application for describing mathematical notation and capturing both its structure and content, MathML is set to make serving, receiving, and processing of mathematics on the Web as achievable as the same operations are with HTML text today.

- SMIL—Synchronized Multimedia Integration Language (<http://www.w3.org/TR/REC-smil>)

Language for specifying interaction of multimedia objects separated in time and space to create an integrated multimedia presentation. Using SMIL (pronounced “smile”), an author can describe the temporal behavior of the presentation and the lay-



out of the objects on the screen, and can associate hyperlinks with multimedia objects.

- P3P—Platform for Privacy Preferences Syntax Specification (<http://www.w3.org/TR/WD-P3P>)

Defines mechanisms: for a user agent (e.g., a web browser) to be automatically informed of a site's data collection and privacy practices; for a user agent and the service to either automatically negotiate an agreement about privacy preferences or for the agent to notify the user and take instruction concerning exchanging information; and for exchange of such data between the user agent and the service as is approved by the user and is consistent with the user's privacy preferences and with any current negotiated agreement.

As well as being an XML application, P3P is an application of RDF.

- RDF—Resource Description Framework Model and Syntax Specification (<http://www.w3.org/TR/REC-rdf-syntax>)

A foundation for processing metadata (i.e., "data about data"), it provides interoperability between applications that exchange machine-understandable information on the Web.

Associated Standards

XML activity has given rise to related activities for using XML, including:

- DOM Level 1—Document Object Model Level 1 Specification (<http://www.w3c.org/TR/1998/REC-DOM-Level-1-19981001>)

A platform- and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of documents. The DOM defines a standard set of objects for representing HTML and XML documents, a standard model for operations on these objects, and a standard interface for manipulating them.

- XML Media Types (<ftp://ftp.isi.edu/in-notes/rfc2376.txt>)

This RFC (Request For Comment) proposes the text/xml and application/xml media types for exchanging XML entities (read "documents and fragments") using HTTP.

Other Proposed Standards and Industry Initiatives

Shared or community tag sets are critical to the success of XML for interchange of information. The following list is a partial list of XML tag sets and efforts to create shared XML tag sets for use in specific industries or with specific types of information. It is through the efforts of these groups that the most expensive, most complex, and arguably most important work in making XML viable is being done.

The list is taken from the XML page of Robin Cover's excellent SGML/XML Web Page at <http://www.oasis-open.org/cover/xml.html>.

- Astronomical Markup Language,
- Bioinformatic Sequence Markup Language (BSML),
- Broadcast Hypertext Markup Language (BHTML),
- CDIF XML-Based Transfer Format,
- Channel Definition Format, CDF (based on XML),
- Chemical Markup Language,
- Coins: Tightly Coupled JavaBeans and XML Elements,
- Cold Fusion Markup Language (CFML),
- Customer Support Consortium,
- DMTF Common Information Model (CIM),
- Document Content Description for XML (DCD),
- Educom Instructional Management Systems Project (IMS),
- Encoded Archival Description (EAD),
- rExtensible Forms Description Language (XFDL),
- rGedML: [GEDCOM] Genealogical Data in XML,
- HTML Threading—Use of HTML in E-mail,
- HTTP Distribution and Replication Protocol (DRP),
- IEEE LTSC XML Ad Hoc Group,
- Information and Content Exchange (ICE),
- InterX.org Initiative,
- Java Help API,
- Java Speech Markup Language (JSML),
- LACITO Projet Archivage de données linguistiques sonores et textuelles [Linguistic Data Archiving Project],
- Legal XML Working Group,
- Meta Content Framework Using XML (MCF),
- Metadata—PICS,
- Newspaper Association of America (NAA)—Classified Ads Format,
- Notes Flat File Format (NFF),
- NuDoc Technology,
- Ontology and Conceptual Knowledge Markup Languages,
- Open Applications Group—OAGIS 6,
- Open Financial Exchange,
- Open Settlement Protocol (OSP)—ETSI/TIPHON,
- Open Software Description Format (OSD),
- Open Trading Protocol (OTP),
- OpenMLS—Real Estate DTD Design,
- OpenTag Markup,
- Process Interchange Format XML (PIF-XML),
- SWAP—Simple Workflow Access Protocol,
- Schema for Object-oriented XML (SOX),
- Scripting News in XML,
- Signed Document Markup Language (SDML),
- Structured Graph Format (SGF),
- Synchronized Multimedia Integration Language (SMIL),
- Telecommunications Interchange Markup (TIM, TCIF/IPI),
- Text Encoding Initiative (TEI),
- The Australia New Zealand Land Information Council (ANZLIC)—Metadata,
- Translation Memory eXchange (TMX),



- Tutorial Markup Language (TML),
- UML eXchange Format (UXF),
- Vector Markup Language (VML),
- Virtual Hyperglossary (VHG),
- VoxML Markup Language,
- W3C Specifications Documentation,
- WAP Wireless Markup Language Specification,
- WDDX—Web Distributed Data Exchange,
- WEBDAV (IETF “Extensions for Distributed Authoring and Versioning on the World Wide Web”),
- Weather Observation Definition Format (OMF),
- Web Collections using XML,
- Web Interface Definition Language (WIDL),
- Web Standards Project (WSP),
- WebBroker: Distributed Object Communication on the Web,
- X-ACT—XML Active Content Technologies Council,
- XLF (Extensible Log Format) Initiative,
- XML Metadata Interchange Format (XMI)—Object Management Group (OMG),
- XML and VRML (Virtual Reality Modeling Language),
- XML for Workflow Management [NIST],
- XML for the Automotive Industry—SAE J2008,
- XML-Data,
- XML-F (“XML for FAX”),
- XML/EDI—Electronic Data Interchange,
- XML/EDI Repository Working Group,
- XSchema,
- vCard Electronic Business Card.

XML and E-Commerce

XML initiatives directly related to e-commerce include:

- OTP—Open Trading Protocol (<http://www.otp.org>)

A consortium with over 30 member companies has developed an XML-based standard for information exchange on the Internet to enable a framework for multiple forms of electronic commerce.
- OFX—Open Financial Exchange (<http://www.ofx.net>)

A unified specification for the electronic exchange of financial data between financial institutions, business, and consumers via the Internet.
- SDML—Signed Document Markup Language (<http://www.w3.org/TR/1998/NOTE-SDML-19980619>)

A generic method for digitally signing a document, one or more sections of a document, and/or multiple documents together, where the signature becomes part of the SDML document and can be verified by recipients as the document travels through the business process.
- BIPS—Bank Internet Payment Service (<http://www.fstc.org/projects/bips/>)

A protocol for sending payment instructions to banks safely over the Internet and a payment server architecture for processing those payment instructions.

- XML/EDI (<http://www.xmledi.net>)

“Provides a standard framework/format to describe different types of data—for example, an invoice, healthcare claim, project status—so that the information, whether in a transaction, catalog, or a document in a workflow, can be searched, decoded, manipulated, and displayed consistently and correctly by implementing EDI dictionaries.”

- ANSI ASC X12/XML

The American National Standards Institute (ANSI) chartered the Accredited Standards Committee (ASC) X12 in 1979 “to develop uniform standards for interindustry electronic interchange of business transactions—electronic data interchange (EDI).” CommerceNet Consortium, XML/EDI Group, and ANSI ASC X12 have entered into a joint project to investigate how to translate ANSI ASC X12 data elements, segments, and transactions into XML.

- ICE—Information and Content Exchange (<http://www.vignette.com>)

“Provides businesses with an XML-based common language and architecture that will facilitate the process of automatically exchanging, updating, supplying, and controlling assets in a trusted fashion (building on OPS/P3P) without manual packaging or knowledge of remote Web site structures.”

- CommerceNet Industry Initiative (<http://www.commerce.net>)

A not-for-profit market and business development organization with a mission to make electronic commerce easy, trusted, and ubiquitous.

- CBL—Common Business Library (<http://www.veosystems.com>)

An extensible, customizable, public collection of XML documents and document components that support generic business concepts.

Relationship of XML to SGML

One of the design goals of the XML effort is that “XML shall be compatible with SGML.” XML is an application profile or restricted form of SGML. All XML documents are conforming SGML documents.

Standard Generalized Markup Language (SGML) is an ISO standard (ISO 8879) for defining non-procedural languages for the marking up of text. Published in 1986, it is widely used by governments, businesses, and publishers. Because of the complexity of the tasks, the number of optional or modifiable features of SGML, and the size of the “typical” user organiza-



tion, SGML has developed, whether earned or not, a fearsome reputation for its difficulty and cost to implement. Like most generalizations, this is true in some cases but not in others; SGML also has many success stories, small-scale users and uses, and free tools. In fact, the Linux HOWTO documentation is being written in SGML and formatted into multiple output formats using free SGML tools (see <http://www.sgmltools.org/>).

XML is an application profile because it supports a subset of the functionality of SGML by choosing which optional features of SGML to include, restricting the grammar of XML DTDs, and hard-wiring aspects, such as character set, that—in a general SGML system—can be different for different documents. However, the subset of SGML that XML represents is not a radical departure from most “mainstream” SGML applications: in practice, some of SGML’s optional features were seldom implemented in tools and even more seldom used, and some aspects of SGML—such as the ability to selectively omit markup—are relics of the lack of tools and high cost of storage when SGML was being designed over 12 years ago, and are not used or are just plain ignored by current SGML applications. When the XML effort started as the W3C SGML Editorial Review Board (later XML Working Group) supported by the SGML Working Group (later XML Special Interest Group) mailing list, consensus was quickly reached on omitting many features of SGML from XML, largely because the omitted features were ones that few people were using in their SGML work.

As an application profile, XML also attaches significance to some constructs that an SGML system would parse but otherwise ignore. For example, XML documents can begin with an “XML Declaration” that has the same format as an SGML processing instruction. If it is present, an XML processor must parse its contents and make use of what it declares, but an equivalent SGML processor would simply pass the contents of the processing instruction through to the following application, just as it would any other processing instruction. In addition, the XML recommendation declares that element and attribute names beginning with “xml” are reserved for standardization, and it defines special meanings for “xml:space” and “xml:lang” attributes, whereas, out of the box, an SGML system would not attach special significance to names beginning with “xml” and would not implement the correct behavior for “xml:space” and “xml:lang” attributes. Since XML documents are conforming SGML documents, XML documents can be processed by an SGML system; but, much like any other SGML application, an SGML system would have to be set up to recognize the special constructs of an XML application.

A major difference between XML and SGML as it was defined in 1986 is the notion of well-formedness and parsing without a DTD. In order for XML to be a subset of SGML, SGML changed to allow this option (although it is called “tag-valid” in SGML) plus imple-

menting a few other changes that kept XML and SGML in sync—some of which had already been on the drawing board for an SGML revision, and some of which hadn’t.

What Can Be Done with XML That Can’t Be Done without XML?

Nothing. But there is a lot that can be done more easily with XML than any other way.

There are a lot of one-off, short-term projects that don’t need XML—tasks where data could be kept as a binary file, as comma-delimited text fields, or as presentational HTML markup—that, if they were large-scale projects, would be better done with XML. When there’s a large amount of data, a large number of people handling the data, or the data will be in use longer than the machine on which it resides, the self-describing nature of the XML markup makes the data easier to manipulate, easier to comprehend, and longer lived.

What Will XML Enable?

- A longer life-span for your information.

XML data is plain text, and the XML markup is just plain text delimited by specific characters. Compare that to word-processor file formats that change every two years or data that exists only in a proprietary database: even when XML is only used as an interchange format, it will give you more freedom to use and re-use your data without being tied to specific hardware or software.

- A richer, more intelligent Web.

The extensibility of XML allows you and, perhaps more importantly, allows entire industry and academic groups, to define a markup that describes the data instead of using HTML markup for making the data look good in a particular brand of browser. The improved markup will lead to improved search engines that can match on tags as well as text content, and will support intelligent manipulation of the data by the client, since the markup isn’t tied to the appearance of the formatted data.

- Improved, ubiquitous, machine-machine communication.

XML is just a data format, but the availability of XML processors for reading and validating XML input and the standardization of the Document Object Model for reading, writing, and manipulating XML data so lowers the barrier to using XML that it is becoming the logical choice for a data format for machine-machine communication.

Instead of developing proprietary, binary data formats, new projects and initiatives are increasingly developing XML models of their data and ex-



changing data marked up with XML. The availability of tools and software libraries for converting between XML markup and the objects or data structures used within a program makes implementing XML much simpler than it otherwise would be, and makes it easy for multiple developers to implement the increasing number of open, XML-based protocols.

The text-based nature of XML data, its self-describing markup, the fact that it can be validated, and the availability of XML processors that can be plugged in to other programs makes XML a natural tool for machine-machine communication. **SV**

© ACM 1067-9936/99/-\$5.00

CNN.com[MAIN PAGE](#)[WORLD](#)[ASIANOW](#)[U.S.](#)[LOCAL](#)[POLITICS](#)[WEATHER](#)[BUSINESS](#)[SPORTS](#)[TECHNOLOGY](#) ↓[computing](#)[personal](#)[technology](#)[SPACE](#)[ENTERTAINMENT](#)[BOOKS](#)[TRAVEL](#)[FOOD](#)[HEALTH](#)[STYLE](#)[IN-DEPTH](#)[custom news](#)[Headline News brief](#)[daily almanac](#)[CNN networks](#)[CNN programs](#)[on-air transcripts](#)[news quiz](#)[CNN WEB SITES:](#)[myCNN.com](#) [CNNi](#)[allpolitics](#) [CNNi](#)[EN ESPAÑOL](#)[em português](#)[SVERSKA](#)[NORGE](#)[danmark](#)[TIME INC. SITES:](#)[Go To ...](#) ▼[MORE SERVICES:](#)[video on demand](#)[video archive](#)[audio on demand](#)[news email services](#)[free email accounts](#)[desktop headlines](#)[pointcast](#)[pagenet](#)[DISCUSSION:](#)[message boards](#)[chat](#)[feedback](#)[sci-tech](#) > [personal technology](#) > [story page](#)

No clicking, no buttons: company offers 'speech sites'

In this story:[Fax it, find it, back up](#)[Lessons from web applied](#)['Speech is one kind of modality'](#)[RELATED STORIES, SITES](#) ↓**INTERACTIVE**

Would you find a "SpeechSite" useful?

Yes, I'd appreciate the
convenience ☐No, I'm not going to call a
computer ☐[View Results](#)[vote](#)**RELATED AUDIO:**Hear samples from
SpeechWorks by ...[E*TRADE](#)[United Airlines](#)[Hewlett Packard](#)[Federal Express](#)

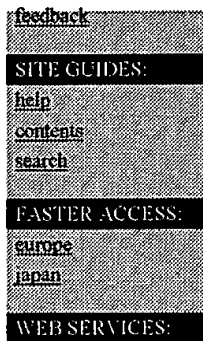
By *Robin Lloyd*
*CNN Interactive Senior
Writer*

(CNN) -- For those who've
burned in voicemail hell, a
Boston company has come
up with a technological
purgatory called speech
sites.

Modeled after the
information and
e-commerce transactions
available on the Web, the
latest in information
technology starts with a telephone number that you'll be
able to dial to reach a computer that hears and guides your
requests and answers them using voice-recognition
technology.

SpeechWorks International has one confirmed SpeechSite
customer so far -- McKesson, a large pharmaceutical
distributor -- with several others on the line. The product
became available in late July.

"SpeechSite brings the Web model of self service to the
telephone," says SpeechWorks Chief Executive Officer



telephone," says SpeechWorks Chief Executive Officer Stuart Patterson. "What we're doing is leveraging and using the Web model."

Several beta speech sites are up and running and should be functional in the fall, he said. SpeechWorks already has limited versions of SpeechSite operating for various companies including E*TRADE, United Airlines, BellSouth and Federal Express.

The idea is to have a computer recognize and answer questions asked by callers, such as where a restaurant is located and its latest reviews or how soon an ordered shipment is expected to arrive. It's not quite like reaching a live person, but it beats voicemail.

Callers will receive recorded voice responses that are more interactive than voicemail and designed to resemble the type of information available at a Web site. The main exception is that speech sites will offer the option of talking to a live operator.

"There's not an exact parallel at Web site for 'I want to talk to somebody now,'" Patterson said. "This is a difficulty we've tried to solve, a core function: let me talk to somebody and no, I don't want to press buttons."

Fax it, find it, back up

The idea with SpeechSite is to change expectations for what can be learned over the telephone, Patterson said, and to make it more like commercial information available on the Internet -- company overviews, press releases, schedules and driving directions.

The technology could change the language people use to get information over a telephone. SpeechSite lingo includes clipped phrases like "fax it," "back up" and "find it," and allows for interruptions which are recognized and processed by the software.

Patterson wants eventually to eliminate voicemail, dial-by-name directories, on-hold frustration and even multiple phone numbers for a single company.

"We saw that the technology is there to blow through to a whole different model for how you interact with the telephone to get through to a company," Patterson said.

The technology is based on SpeechWorks' main product, speech recognition systems. But it also is capable of

"conversation management" -- responding more intelligently to questions and menu choices, Patterson said.

"If you say something out of bounds, we want our next prompt to rein you in. Good systems do not just repeat what they said; they have a different prompt. Good systems will bail you out into a person," he said.

Lessons from web applied

Patterson says the technology will catch on because people have "had it up to their eyeballs with touch tone" and the Web has raised consumers' expectations for self-service.

"The idea of how you relate to customers revolves around e-business, of which the first modality is the Web. That fits into speech sites. It has given us all the lessons of what you need -- we combine information with transactions and communications."

Patterson is confident that companies will sign up to get their company configured for a speech site -- at a price of \$50,000 to \$150,000.

The plan is a guarantee that 98 percent of callers will get the information they call for, Patterson said, and responses arrive in less than 2 seconds.

Within nine months, Patterson envisions the equivalent of external hot links -- callers can move from company to company simply by stating names.

'Speech is one kind of modality'

SpeechSite is based on technology initially developed and still being advanced at the Massachusetts Institute of Technology. One of SpeechWorks' founders worked with Victor Zue, who runs a highly regarded speech recognition research group at MIT's Laboratory for Computer Science.

Jim Glass, a principal research scientist in Zue's MIT lab, says speech recognition technologies have their pluses and minuses.

"Speech is one kind of modality. If I happen to have a computer display in front of me in a noisy environment, I'll want to use my mouse and click," he said.

"But if I'm not near a computer then probably accessing it by voice is a very convenient way to access this information. Certainly speech and language is a very natural thing for humans."

Zue's group now is working on giving people more license in the questions they ask than SpeechSite is likely to offer, Glass said. But it's hard to make those systems recognize questions phrased any way that a user might like.

For a taste, give a call to MIT's Jupiter system, which gives interactive worldwide weather reports around the clock. The number is (888) 573-8255. To check out SpeechWorks' speech site, dial (617) 428-4444.

RELATED STORIES:

[Phones to boost e-commerce?](#)

June 14, 1999

[IBM offers speech extension to XML](#)

February 19, 1999

[PCs still learning to listen](#)

December 10, 1998

[Speech technology is everywhere](#)

December 3, 1998

[Look, Ma, no keyboard](#)

November 19, 1998

RELATED SITES:

[SpeechWorks International](#)

[MIT Laboratory for Computer Science](#)

Note: Pages will open in a new browser window

External sites are not endorsed by CNN Interactive.

LATEST HEADLINES:

SEARCH CNN.com

Enter keyword(s)

go help

[Back to the top](#)

© 2001 Cable News Network. All Rights Reserved.
Terms under which this service is provided to you.
Read our [privacy guidelines](#).

PCWORLD.COM

TECHNOLOGY ADVICE YOU CAN TRUST

Home

News

Reviews

How-To

Downloads

Tools

Product Finder

Magazine

Search for:

SEARCH

Advanced Search

BROWSE BY TOPIC

USE FIND.PCWORLD.COM

PCs Still Learning to Listen

Technology Outlook attendees say better connectivity will hasten consumer speech recognition products.

David Needle, special to PC World
Tuesday, December 08, 1998

If home appliances of the future could complain, one lament may echo the old song, "Everybody's Talking at Me." The number of speech-enabled devices will grow rapidly in the next few years, say a panel of experts at the Technology Outlook conference in San Francisco.

"Ultimately, you want [to connect] devices all over the house," says William Meisel, publisher of *Speech Recognition Update*. Meisel suggested that a user could be in the bedroom and tell a speech recognition device to record that evening's football game. Or that same household's television might report that the stove has finished cooking dinner.

Business users can expect similar connectivity.

"Today you have PDAs, Palm Pilots, cell phones, and other devices," says Ronald Croen, chief executive officer of Nuance Communications. "Instead of thinking of what you can do with one of these devices, think of what you can do in a connected environment where transactions can be dynamic." Nuance, in Menlo Park, California, develops a speech recognition software platform for telephone transactions and network applications.

Speech recognition has the greatest short-term potential in devices that people are already used to talking to, such as the telephone, commented panelists and others at the conference. Existing voice recognition services let users retrieve e-mail and stock quotations over the telephone. In the future, speech recognition could address the need to replace the telephone's "terrible touch tone interface," noted Meisel.

To lure PC users, speech recognition advocates face the challenge of replacing the keyboard and mouse, "which works for most people," notes Croen. "The PC is not really broken."

But National Semiconductor Chief Executive Officer Brian Halla drew applause later in the day when he demonstrated Windows-based voice recognition software from Lernout & Hauspie, a Belgian company with offices in Burlington, Massachusetts. The audience heard Halla begin his address offstage as he spoke into a microphone connected to a PC. Halla used a series of short voice instructions to launch the Yahoo Web site, find an article in a back issue of *Time* magazine, and insert sections of the article into his PowerPoint presentation.

Microsoft Slow on the Draw

Although Microsoft has an investment in Lernout & Hauspie, Chief Executive Officer Bill Gates doesn't seem to think speech recognition is ready for the mainstream.

"A year ago, 90 percent of the people who bought [speech recognition programs] didn't use them," Gates said in an interview last month at Comdex.

"That means 10 percent of those people found value in the product. How do you get that number [of users] up to 50 or 60 percent?" asked Gates. "You need a big breakthrough. I can't say when it will happen, but we're working on it." Like the true software guy he is, Gates said the biggest impediment to better speech recognition is hardware: "We need much better microphones."

Ironically, Gates thought the small percentage of satisfied users is an indication of the technology's potential.

One frustrated potential customer is Richard Norman, president of Hyperchip.

"My company would pay thousands of dollars for continuous speech recognition [that worked]," says Norman. "But the desktop products I've tried have been terrible."

Microsoft is using speech recognition in non-PC devices such as a new Microsoft-branded telephone.

But one competitor, Todd Mozer, chief executive officer of Sensory, in Sunnyvale, California, says Microsoft's efforts are too complicated because the products must also be connected to a PC. Sensory develops speech-recognition applications for consumer devices and telephone handsets.

As telephone companies add speech recognition to their back-end systems, the potential number of users is staggering, according to Meisel. "You will be able to upgrade a hundred million users [with compatible telephones]," says Meisel, "who don't have to download anything or worry about their phone crashing."

[Printer Friendly Version](#)

MARCH 13, 2003

RELATED ARTICLES

[Top 10 Hard Drives](#)

[Nuance Communications](#)

[Lernout & Hauspie](#)

[Sensory](#)

[Speech Recognition Makers
Promise "Assistants"](#)

[See all related items](#)

RELATED FILES

[Internet Tools: Publish a blog,
manage your site, record Web
video](#)

[Free Utilities To Boost Your
Browser](#)

[See all related downloads](#)

PC CHECKUP

Tests to run:

[Speed Check](#)

[Disk Health](#)

[Virus Scan](#)

[See all tools](#)

PRODUCT GUIDES

[Desktops](#)

[Notebooks](#)

[Printers](#)

[Monitors](#)

[See all product guides](#)

FREE NEWSLETTERS

Get our weekly news recap or daily
download digest.

☐ [Weekly Brief](#)

☐ [Hot Shareware](#)

Enter your e-mail:

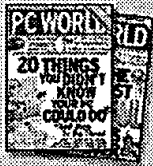
[See all newsletters](#)

MOBILE COMPUTING

James Martin helps
you make the most
of your computing
on the go.

[See all Mobile
Computing](#)





**Special
Anniversary
Offer: Only
\$1/issue!**

Receive 2 RISK-FREE Issues of PC World and 10 FREE Power Guides!

Enter your trial subscription and you'll receive 2 Risk-Free Issues plus 10 FREE Instant Power Guides and Ebooks CD-ROM. If you like PC World, pay just \$20 for 18 more issues (20 in all). Otherwise, write "cancel" on the bill, return it, and owe nothing.

BONUS OFFER! Pay now and get 2 Extra Issues FREE! That's 22 Issues for the same low price! [click here](#)

Try PC World Risk-Free, just fill in the form and click Submit!

Name	City	
<input type="text"/>	<input type="text"/>	
Address 1	State	Zip Code
<input type="text"/>	<input type="text"/>	<input type="text"/>
Address 2	E-Mail	
<input type="text"/>	<input type="text"/>	
<input type="button" value="Submit"/>		

☐ Canadian residents, [click here](#) | All other foreign residents, [click here](#)



[About Us](#) | [Contact Us](#) | [Advertise](#) | [Corrections](#) | [Subscribe to the Magazine](#) | [Member Services](#)
[Site Map](#) | [Copyright & Permissions](#) | [Terms of Service Agreement](#) | [ASAP Guidelines](#) | [Privacy Statement](#)